Claims

1. (Currently amended) A computer-readable medium storing computer-executable instructions for causing a computer system programmed thereby to perform a method comprising ~~A method for assessing a relationship of an interface to a unit, wherein a software program comprises plural units, and wherein a unit exposes one or more interfaces, the method comprising~~:

detecting a reference to an interface ~~of a unit of software~~, wherein a software program comprises plural units, and wherein one of the plural units exposes the interface;

determining if the ~~unit that exposes the~~ interface ~~is known~~ has been wrapped by checking a data structure that tracks interface wrapping for the plural units of the software program ~~comprising one or more entries, wherein an entry maps an interface to a unit identity~~;

if the interface has been wrapped, returning a reference to an existing entry for the interface from the data structure, wherein the existing entry associates the interface with a unit identity for the unit that exposes the interface;

if the ~~unit that exposes the~~ interface ~~is not known~~ has not been wrapped,

discovering the unit identity of the unit that exposes the interface;

adding ~~an~~ a new entry to the data structure, wherein the new entry ~~maps~~ associates the interface ~~to~~ with the discovered unit identity; and

returning a reference to the new entry.

~~performing an operation based upon an entry of the data structure.~~

2. (Canceled)

3. (Currently amended) The computer-readable medium ~~method~~ of claim 1 wherein the step of detecting comprises:

noting one or more return parameters from a called function; and

parsing the one or more return parameters to detect [[a]] the reference to ~~an~~ the interface.

4. (Currently amended) The computer-readable medium ~~method~~ of claim 3 wherein the called function is a unit creation function.

5. (Currently amended) The computer-readable medium ~~method~~ of claim 1 wherein the step of detecting comprises:

noting one or more outgoing parameters to a called function; and

parsing the one or more outgoing parameters to detect [[a]] the reference to ~~an~~ the interface.

6. (Currently amended) The computer-readable medium ~~method~~ of claim 1 wherein the data structure ~~is~~ comprises a hash table, and wherein the step of determining comprises:

hashing the detected reference~~;~~

~~if the detected reference hashes to a unit identity, returning a value that indicates the unit identity of the unit that exposes the interface is known; and~~

~~if the detected reference does not hash to a unit identity, returning a value that indicates the unit identity of the unit that exposes the interface is not known~~.

7. (Currently amended) The computer-readable medium ~~method~~ of claim 1 wherein the data structure ~~is~~ comprises a hash table, wherein the step of adding ~~an~~ the new entry comprises:

creating [[a]] the new entry in the hash table, wherein the new entry associates the interface with the discovered unit identity.

8. (Currently amended) The computer-readable medium ~~method~~ of claim 1 ~~wherein an interface wrapper stores data comprising a unit identity,~~ wherein the data structure ~~is~~ comprises a hash table for associating interfaces with interface wrappers, wherein the existing entry is for an existing interface wrapper in the hash table, and wherein the step of determining comprises:

hashing the detected reference~~;~~

~~if the detected reference hashes to an interface wrapper, returning a reference to the interface wrapper; and~~

~~if the detected reference does not hash to an interface wrapper, returning a value indicating that the interface does not have an interface wrapper~~.

9. (Canceled)

10. (Currently amended) The computer-readable medium ~~method~~ of claim 1 ~~wherein an interface wrapper stores data comprising a unit identity,~~ wherein the data structure ~~is~~ comprises a hash table for associating interfaces with interface wrappers, and wherein the new entry is for a new interface wrapper that stores the unit identity. ~~the method further comprising:~~

~~if the unit that exposes the interface is not known,~~

~~creating an interface wrapper, wherein the interface wrapper stores data comprising the discovered unit identity; and~~

~~during the step of adding an entry, creating a new entry in the hash table, wherein the new entry associates the interface with the created interface wrapper.~~

11. (Currently amended) The computer-readable medium ~~method~~ of claim 1 wherein a local variable stores data comprising the unit identity of the unit that exposes the interface ~~from which the detected reference originated~~, and wherein the step of discovering the unit identity comprises noting the value stored in the local variable.

12. (Currently amended) The computer-readable medium ~~method~~ of claim 11 wherein an instrumentation system provides the unit identity of the unit that exposes the interface ~~from which the detected reference originated~~.

13. (Currently amended) The computer-readable medium ~~method~~ of claim 1 wherein the method further comprises ~~comprising~~:

if the ~~unit that exposes the~~ interface ~~is known~~ has been wrapped,

verifying the unit identity of the unit that exposes the interface; ~~and~~

~~overwriting an entry of the data structure, wherein the entry maps the interface to the verified unit identity~~.

14. (Currently amended) The computer-readable medium ~~method~~ of claim 1 ~~wherein the step of performing an operation upon an entry of the data structure comprises~~ wherein the method further comprises:

detecting a communication passing through the interface;

measuring the size of the communication;

determining the unit identity of the unit that exposes the interface from the returned reference to the existing or new entry of the data structure; and

associating the measured size with the unit that exposes the interface.

15. (Currently amended) The computer-readable medium ~~method~~ of claim 1 ~~wherein the step of performing an operation based upon an entry of the data structure comprises~~ wherein the method further comprises:

receiving from a client unit a call to a unit activation function for a unit to be activated among the plural units ~~from a client unit~~;

determining [[the]] a unit identity of the client unit from ~~the entry of~~ the data structure;

classifying the unit to be activated based upon the unit identity of the client unit, resulting in a classifier;

determining a location in a distributed computing environment using the classifier; and

routing the call to the location.

16. (Currently amended) A computer-readable medium storing computer-executable instructions for causing a computer system programmed thereby to perform a method comprising ~~A method for assessing a relationship of an interface to a unit, wherein a software program comprises plural units, and wherein a unit exposes one or more interfaces, the method comprising~~:

on receiving a reference to an interface as a return parameter from a function call,

determining if ~~the unit that exposes~~ the interface is ~~known~~ wrapped by checking a data structure that tracks interface wrapping for plural components of software, wherein one of the plural components exposes the interface ~~comprising one or more entries, wherein an entry maps an interface to a unit identity~~;

if ~~the unit that exposes~~ the interface is not ~~known~~ wrapped,

discovering ~~the unit~~ component identity of the ~~unit~~ component that exposes the interface; and

adding ~~an~~ a new entry to the data structure, wherein the new entry ~~maps~~ associates the interface ~~to~~ with the discovered ~~unit~~ component identity~~; and~~

~~performing an operation based upon an entry of the data structure~~.

17. (Canceled)

18. (Currently amended) The computer-readable medium ~~method~~ of claim 16 wherein the data structure ~~is~~ comprises a hash table, and wherein the step of determining comprises:

hashing the ~~detected~~ received reference;

~~if the detected reference hashes to a unit identity, returning a value that indicates the unit identity of the unit that exposes the interface is known; and~~

~~if the detected reference does not hash to a unit identity, returning a value that indicates the unit identity of the unit that exposes the interface is not known~~.

19. (Currently amended) The computer-readable medium ~~method~~ of claim 16 wherein the data structure ~~is~~ comprises a hash table, wherein the step of adding ~~an~~ the new entry comprises:

creating [[a]] the new entry in the hash table, wherein the new entry associates the interface with the discovered ~~unit~~ component identity.

20. (Currently amended) A computer-readable medium storing computer-executable instructions for causing a computer system programmed thereby to perform a method comprising ~~A method for assessing a relationship of an interface to a unit of software using an interface wrapper, wherein the software comprises plural units, and wherein a unit of software exposes one or more interfaces, the method comprising~~:

detecting a reference to an interface, wherein one of plural components of software exposes the interface ~~of a unit of software~~;

determining if the interface is wrapped using a hash table ~~for associating an interface with an interface wrapper~~;

if the interface is wrapped, providing to a client component a reference to an existing interface wrapper, wherein the existing interface wrapper stores a reference to instrumentation, the reference to the interface, and component identity of the component that exposes the interface;

if the interface is not wrapped,

creating [[an]] a new interface wrapper for the interface, wherein the new interface wrapper stores [[a]] the reference to the instrumentation and the reference to the interface;

creating a new entry in the hash table, wherein the new entry associates the interface with the created new interface wrapper;

discovering the ~~unit~~ component identity of the ~~unit~~ component that exposes the interface;

storing in the new interface wrapper the ~~unit~~ component identity of the ~~unit~~ component that exposes the interface; and

providing to [[a]] the client ~~unit~~ component a reference to the new interface wrapper; and

receiving from the client ~~unit~~ component an invocation of the instrumentation through the provided reference to the existing or new interface wrapper.

21. (Canceled)

22. (Currently amended) The computer-readable medium ~~method~~ of claim 20 wherein the step of determining ~~whether~~ if the interface is wrapped comprises:

hashing the detected reference;

if the detected reference hashes to [[an]] the existing interface wrapper, returning [[a]] the reference to the existing interface wrapper; and

~~if the detected reference does not hash to an interface wrapper~~ otherwise, returning a value that indicates the interface is not wrapped.

23. (Currently amended) The computer-readable medium ~~method~~ of claim 20 wherein the reference to instrumentation comprises a pointer to a table comprising at least one pointer to one or more instrumentation functions.

24. (Currently amended) The computer-readable medium ~~method~~ of claim 20 wherein the step of creating [[an]] the new interface wrapper further comprises storing in the new interface wrapper a type description of the interface.

25. (Currently amended) The computer-readable medium ~~method~~ of claim 20 wherein a local variable stores the ~~unit~~ component identity of the ~~unit~~ component that exposes the interface

~~from which the detected reference originated~~, and wherein the step of discovering the component identity of the ~~unit~~ component that exposes the interface comprises noting the value stored in the local variable.

26. (Currently amended) The computer-readable medium ~~method~~ of claim 20 wherein the invocation comprises a communication from the client ~~unit~~ component directed towards the ~~unit~~ component that exposes the interface, the method further comprising:

measuring the size of the communication using the instrumentation;

associating the measured size with the ~~unit~~ component that exposes the interface and the client ~~unit~~ component; and

calling the ~~unit~~ component that exposes the interface.

27. (Currently amended) The computer-readable medium ~~method~~ of claim 26 wherein the method further comprises ~~comprising~~:

before the step of calling the ~~unit~~ component, setting a return address for returning from the called ~~unit~~ component as a reference to the instrumentation;

after the step of calling the ~~unit~~ component, receiving from the ~~unit~~ component that exposes the interface a second invocation of the instrumentation, wherein the second invocation comprises a second communication from the ~~unit~~ component to the client ~~unit~~ component;

measuring the size of the second communication using the instrumentation;

associating the measured size of the second communication with the ~~unit~~ component that exposes the interface and the client ~~unit~~ component; and

returning control to the client ~~unit~~ component.

28. – 56. (Canceled)

57. (Currently amended) A computer-readable medium storing computer-executable instructions for causing a computer system programmed thereby to perform a method comprising ~~A method for handling an undocumented interface between a client unit and a server unit of an application program, wherein the interface comprises one or more functions, and wherein a description file comprises a description of the interface, the method comprising~~:

receiving from a client unit a call to a function of an interface;

determining if the interface is documented based upon [[the]] description of [[the]] a description file;

if the interface is undocumented,

transferring execution to the function, wherein the function performs an operation; and

after the function completes,

transferring execution to the client unit.

58. (Canceled)

59. (Currently amended) The computer-readable medium ~~method~~ of claim 57 wherein a call stack holds one or more parameters of the call to the function pushed by the client unit, and wherein a stack pointer points to the top of the stack, the method further comprising:

if the interface is undocumented, before the step of transferring execution to the function, storing the value of the stack pointer;

after the function completes, before the step of transferring execution to the client unit,

popping the one or more parameters from the call stack;

calculating the size of the one or more parameters based upon the stored value of the stack pointer and the current value of the stack pointer; and

storing the calculated size in the description file.

60. (Currently amended) The computer-readable medium ~~method~~ of claim 57 wherein a call stack holds a return address for the client unit pushed by the client unit, the method further comprising:

if the interface is undocumented, before the step of transferring execution to the function,

storing the return address for the client unit;

replacing the return address for the client unit on the call stack with a return address for an instrumentation function;

after the function completes, before the step of transferring execution to the client unit,

popping the return address from the call stack; and

transferring execution to the instrumentation function.

61. (Currently amended) The computer-readable medium ~~method~~ of claim 57 wherein a call stack holds one or more parameters of the call to the function pushed by the client unit, wherein the call stack holds a return address for the client unit pushed by the client unit, and wherein a stack pointer points to the top of the stack, the method further comprising:

if the interface is undocumented, before the step of transferring execution to the function,

storing the value of the stack pointer;

storing the return address for the client unit;

replacing the return address for the client unit on the call stack with a return address for an instrumentation function;

after the function completes, before the step of transferring execution to the client unit,

popping the return address from the call stack; and

popping the one or more parameters from the call stack;

transferring execution to the instrumentation function;

calculating the size of the one or more parameters based upon the stored value of the stack pointer and the current value of the stack pointer; and

storing the calculated size in the description file.

62. (Canceled)

63. (Currently amended) The computer-readable medium ~~method~~ of claim 57 wherein the step of determining if the interface is documented comprises:

examining the description file; and

if the description file contains no description, indicating that the interface is undocumented.

64. (Currently amended) The computer-readable medium ~~method~~ of claim 57 wherein the step of determining if the interface is documented comprises:

examining the description file; and

if the description file contains inadequate information for parsing a communication across the interface, indicating that the interface is undocumented.

65. (Currently amended) The computer-readable medium ~~method~~ of claim 57 wherein a data structure stores an identifier for the client unit and an identifier for [[the]] a server unit, the method further comprising:

if the interface is undocumented,

noting the relationship of the undocumented interface to the client unit and the server unit based upon the identifiers; and

noting a pair-wise location constraint between the client unit and the server unit.

66. (Canceled)

67. (New) The computer-readable medium of claim 1 wherein the computer-executable instructions are for automatic distributed partitioning system software.

68. (New) The computer-readable medium of claim 1 wherein the reference to the interface is a pointer to the interface.

69. (New) The computer-readable medium of claim 16 wherein the computer-executable instructions are for automatic distributed partitioning system software.

70. (New) The computer-readable medium of claim 16 wherein the reference to the interface is a pointer to the interface.

71. (New) The computer-readable medium of claim 20 wherein the computer-executable instructions are for automatic distributed partitioning system software.

72. (New) The computer-readable medium of claim 20 wherein the reference to the interface is a pointer to the interface.

73. (New) The computer-readable medium of claim 57 wherein the computer-executable instructions are for automatic distributed partitioning system software.